Wireless Networks 9, 131–142, 2003

Integrating Distributed Channel Allocation and Adaptive Handoff Management for QoS-Sensitive Cellular Networks

GUOHONG CAO
Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA 16802, USA

Abstract. Next generation high-speed cellular networks are expected to support multimedia applications, which require QoS provisions. Since frequency spectrum is the most expensive resource in wireless networks, it is a challenge to support QoS using limited frequency spectrum. In the literature, two orthogonal approaches are used to address the bandwidth utilization issue and the QoS provision issue; that is, channel allocation schemes have been proposed to improve bandwidth efficiency, whereas handoff management schemes, based on bandwidth reservation, have been proposed to guarantee a low connection dropping rate. However, little effort has been taken to address both issues together. In this paper, we integrate distributed channel allocation and adaptive handoff management to provide QoS guarantees and efficiently utilize the bandwidth. First, we present a complete distributed distributed channel allocation algorithm and propose techniques to reduce its message complexity and intra-handoff overhead. Second, we integrate the proposed distributed channel allocation algorithm with an adaptive handoff management scheme to provide QoS guarantees and efficiently utilize the bandwidth. Detailed simulation experiments are carried out to evaluate the proposed methodology. Compared to previous schemes, our scheme can significantly reduce the message complexity and intra-handoff overhead. Moreover, the proposed scheme can improve the bandwidth utilization while providing QoS guarantees.

Keywords: distributed channel allocation, handoff management, call dropping, quality of service, intra-handoff, cellular networks

1. Introduction

Next generation high-speed cellular networks are expected to support multimedia applications (video, voice, and data). As such, it is important that these networks provide quality-of-service (QoS) guarantees. Due to user mobility, a communication session may be dropped if the destination cell does not have enough bandwidth to support the connection. The connection dropping rate (CDR) is a very important QoS parameter, since it is very disturbing for a user to have a dropped connection. One solution to reduce the CDR is by reserving enough bandwidth at the destination cell. However, if this bandwidth reservation is not carefully designed, some bandwidth may be wasted. Since the bandwidth (frequency spectrum) is the most expensive resource in wireless networks, it is a challenge to support QoS using limited spectrum. In the literature, two orthogonal approaches are used to address the bandwidth utilization issue and the QoS provision issue; that is, channel allocation schemes have been proposed to improve bandwidth efficiency, whereas handoff management schemes, based on bandwidth reservation, have been proposed to guarantee a low connection dropping rate. However, little effort has been taken to address both issues together. The proposed research will bridge the gap by designing and evaluating distributed channel allocation and handoff management schemes which can provide QoS guarantees and efficiently utilize bandwidth. Before looking at our approach, let's first look at previous work in these two areas.

1.1. Channel allocation algorithms

Many channel allocation algorithms [7,10,12,14,15,22] have been proposed during the last thirty years for cellular networks to avoid channel interference and efficiently utilize the limited frequencies. These algorithms can be classified into three types [15]: fixed, flexible, and dynamic. Among them, dynamic channel allocation (DCA) strategies have been the focus of recent research [2,9,20]. With DCA strategies, a cell [12] may use any channel that will not cause channel interference. Typically, each channel is associated with a priority; when a cell needs a channel, it picks the available channel which has the highest priority. All these algorithms [12,14,15], which are referred to as *centralized* channel allocation algorithms, rely on a *mobile switching center* (MSC) to accomplish channel allocation. More specifically, each cell notifies the MSC when it acquires or releases a channel so that the MSC knows which channels are available in each cell at any time and assigns the available channels to requesting cells accordingly.

Recently, distributed channel allocation algorithms [4,5,9, 13,21] have received considerable attention because of their high reliability and scalability. In this approach, a base station (BS) communicates with other BSs directly to find the available channels and to guarantee that assigning a channel does not cause interference with other cells. In general, a channel allocation algorithm [5] consists of two parts: a *channel acquisition* algorithm and a *channel selection* algorithm. The channel acquisition algorithm is responsible for collecting information from other cells and making sure that cells will not use the channels that can cause interference. The

channel selection algorithm is used to choose a channel from a large number of available channels in order to achieve better channel reuse. Due to the popularity of DCA allocation strategies, most of the distributed channel allocation algorithms use DCA strategies as their channel selection algorithm.

1.2. Handoff management schemes

As distributed channel allocation schemes are designed to improve bandwidth utilization, handoff management schemes are proposed to provide QoS guarantees. Early handoff management schemes [15] are proposed to limit the probability of forced call termination based on handoff prioritization; that is, reserving a fixed or dynamically adjustable number of channels exclusively for handoffs. Other prioritizing schemes [15] allow either the handoff to be queued or new connections to be queued until new channels are obtained in the cell. Recent handoff management schemes [1,6,17–19,23] predict the moving direction of mobile terminals (MTs) and only reserve channels at the potential destination cells instead of at all neighbors. Lee [17] suggested bandwidth reservation depending on the existing connections in adjacent cells. A shadow cluster concept was suggested in [18] to estimate future resource requirements and perform admission control in order to limit the handoff dropping probability. This scheme is based on the precise knowledge of each user mobility in terms of location and time. Lu and Bharghavan [19] explored mobility estimation for an indoor wireless system based on both mobile-specific and cell-specific observation histories. Choi and Shin [6] estimated mobile's handoff behaviors based on a history of observations in each cell. Their scheme is based on the observation that the handoff behavior of an MT is probabilistically similar to the MTs which came from the same previous cell, and now belong to the current cell.

Most of the previous handoff management schemes have been studied and evaluated independently of the channel allocation schemes. As a result, although some handoff management schemes can provide good QoS guarantees, they may not use bandwidth efficiently. Moreover, these handoff management schemes assume that a fixed channel allocation algorithm is used. Due to the difference between fixed and distributed channel allocation schemes, previous channel allocation and handoff management schemes need to be redesigned and re-evaluated.

1.3. Contributions of this paper

In this paper, we integrate distributed channel allocation and adaptive handoff management to provide QoS guarantees and efficiently utilize bandwidth. First, we integrate distributed acquisition algorithms and channel selection algorithms to get a complete distributed channel allocation algorithm. During the integration, we propose schemes to reduce the message complexity and the intra-handoff (handoff within a cell) overhead. Second, we integrate the proposed distributed channel allocation algorithm with an adaptive handoff management scheme to provide QoS guarantees and efficiently utilize the bandwidth. Detailed simulation experiments are carried out to evaluate the proposed methodology. Compared to previous schemes, our scheme can significantly reduce the message complexity and intra-handoff overhead. Moreover, the proposed scheme can improve the bandwidth utilization while providing QoS guarantees.

The rest of the paper is organized as follows. Section 2 presents the system model. In section 3, we propose a complete distributed channel allocation algorithm and propose schemes to reduce its message complexity and inter-handoff overhead. Section 4 integrates the distributed channel allocation algorithm with an adaptive handoff management scheme to guarantee QoS provisions and efficiently utilize the bandwidth. In section 5, we present our simulation results. Section 6 concludes the paper.

2. System model

Most channel selection strategies [2,9] require a *priori* on channel status in order to achieve better channel reuse. For instance, in the dynamic resource acquisition scheme [20], each cell is allocated a set of "nominal" channels before hand; in the geometric strategy [2], each cell must know its "first-choice" channels prior to any channel acquisition. We call the process of assigning special status to channels as *resource planning* [9].

Resource planning

1. Partition the set of all cells into a number of disjoint subsets $G_0, G_1, \ldots, G_{k-1}$, such that any two cells in the same subset are apart by at least a distance of $D_{\min}$. Accordingly, partition the set of all channels into k disjoint subsets: $P_0, P_1, \ldots, P_{k-1}$.
2. The channels in P_i $(i = 0, 1, \ldots, k-1)$ are *primary channels* of cells in G_i, and *secondary channels* of cells in G_j $(j \neq i)$.
3. A cell requests a secondary channel only when no primary channel is available.

For convenience, we say that a cell C_i is a *primary (secondary)* cell of a channel r if and only if r is a primary (secondary) channel of C_i. Thus, the cells in G_i are primary cells of the channels in P_i and secondary cells of the channels in P_j $(j \neq i)$.

Definition 1. Given a cell C_i, the set of *interference neighbors* of C_i, denoted by IN_i, is:

$$IN_i = \{C_j \mid distance(C_i, C_j) < D_{\min}\}.$$

Definition 2. For a cell $C_i \in G_p$ and a channel $r \in P_p$, the *interference primary cells* of r relative to C_i, denoted by $IP_i(r)$, are the cells which are primary cells of r and interference neighbors of C_i; i.e., $IP_i(r) = G_p \cap IN_i$. $IP_i(r)$ is also referred to as an *interference partition subset* of C_i, relative to r.

In order to achieve better channel reuse, each subset G_i should contain as many cells as possible. Then, k should be as small as possible. How to partition the cells is orthogonal to our discussion, but we require that the partition satisfies the following two properties, which have been proved to be the necessary conditions for any *optimal partition method* in [9]:

Property 1. $\forall C_i, C_j \in G_p$: $distance(C_i, C_j) \geqslant D_{\min}$.

Property 2. $\forall C_i, C_j$: $C_i \in IN_j \Rightarrow \forall r (IP_i(r) \cap IP_j(r) \neq \emptyset)$.

Property 1 is obvious. Property 2 is explained as follows: assume a cell C_i is an interference neighbor of C_j; if C_i and C_j request the same channel r, they have at least one common cell which is an interference primary cell of r.

Figure 1 shows one partition, which divides the cells into nine subsets $G_A, G_B, \ldots, G_I$. Cells in $G_A = \{C_{A_i} \mid 0 \leqslant i \leqslant 8\}$ can use the same channel without interference. If two interference neighbors C_{G_2} and C_{D_5} request a primary channel of a cell in G_A, they have a common interference primary cell C_{A_5}. Since the distance between any two nearest cells in a subset is exactly $D_{\min}$, it is an optimal partition in the sense that each channel is maximally reused by its neighbors.

2.1. Handoff and intra-handoff

An MT may cross the boundary between two cells while being active. When this occurs, the necessary state information must be transferred from its previous BS to the BS in the new cell. This process is known as *handoff* (or *inter-handoff*). During a handoff, an MT releases its current channel to its previous BS and is assigned a channel by the new BS.

To achieve better channel reuse, *intra-handoff* (or a channel switch) may be necessary [9]. In an intra-handoff operation, an MT releases its current channel and is assigned a new channel within the same cell. The motivation behind intra-handoff can be understood by an example. In figure 1, suppose cell C_{A_1} has two primary channels $r1$ and $r2$. C_{A_1} is using $r2$, while cells C_{A_2}, C_{A_4}, and C_{A_5} are using $r1$. Even though $r1$ is available in C_{A_1} and $r2$ is available in cells C_{A_2}, C_{A_4}, and C_{A_5}, neither $r1$ nor $r2$ can be borrowed by C_{H_1}. If an intra-handoff is performed, i.e., C_{A_1} releases $r2$ and uses $r1$, C_{H_1} can borrow $r2$.

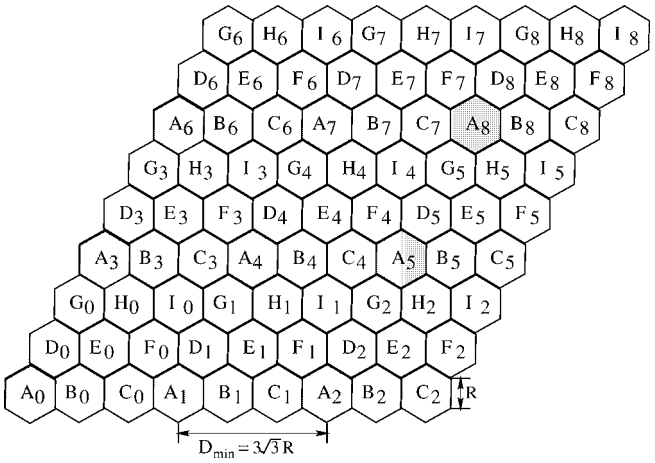

Figure 1. An optimal partition.

3. An adaptive distributed channel allocation algorithm

In this section, we first present a distributed channel acquisition algorithm, and a channel selection algorithm. Then, we integrate them to get a complete distributed channel allocation algorithm. We also present techniques to reduce the message complexity and intra-handoff overhead.

3.1. The channel acquisition algorithm

In our distributed fault-tolerant channel acquisition algorithm, when a cell C_i needs a channel, it does the follows. If it has an available primary channel r, it marks r as an *used channel* and uses r immediately; otherwise, C_i changes to *search mode* and sends a *request* message to each cell in IN_i. When a cell C_j receives the *request* from C_i, if C_j is not in search mode or C_j is in search mode but C_j's request has higher timestamp [16] than C_i's, C_j sends a *reply* message which appends the information about its used channels to C_i; otherwise, C_j defers the *reply*. After the borrower has received *reply* messages from cells in IN_i or the timer timeouts, the borrower computes the available channels and picks one according to the underlying channel selection algorithm. The borrower has to confirm the selected channel with the lenders, since a lender may assign that channel to a new call immediately after it sends out a *reply* message. If each lender responds with an *agree* message, the borrower can use the borrowed channel; otherwise, it picks another available channel and confirms again. If there is no available channel left, the call request is failed. After a channel has been borrowed, the lender marks the channel as an *interference channel*, and will not use it until it is returned by all borrowers.

A formal description of the algorithm is given in figure 2. Two types of control messages are used to acquire the information on available channels: *request* and *reply*. If two or more cells in each other's interference neighbors request the same channel concurrently, a conflict arises. If there is no conflict, three types of messages are exchanged among MSSs to confirm or return a channel: *confirm*, *agree*, and *release*. If there is a conflict, two additional messages are needed: *abort* and *conditional_agree*. Control messages are timestamped using Lamport's clock [16] to determine the request priorities. The solution to conflicts is shown in step D.3. By maintaining $I_i(r)$ and $CI_i(r, j)$, a cell C_i never grants concurrent requests for the same channel from cells within $D_{\min}$. In other words, if two cells, which are separated by a distance less than $D_{\min}$, request the same channel, they will not receive *agree* messages from the same interference primary cell.

Besides conflict resolution, the adoption of *conditional_agree* messages can also avoid wasting available channels, which can be explained by an example. In figure 1, assume that C_{H_4} and C_{F_4} concurrently request channels. Suppose C_{H_4}'s request has smaller timestamp. If there is one

Data structures.

- IN_i, $IP_i(r)$: defined before.
- B_i: a set of channels that C_i can borrow.
- S_i: a set of channels that C_i attaches with its *reply*.
- P_i: the set of primary channels assigned to C_i.
- U_i: the set of used channels at C_i. U_i is initialized to empty.
- $I_i(r)$: the set of cells to which C_i has sent an *agree*(r) message. If $I_i(r) \neq \emptyset$, r is an interference channel of C_i. Then, C_i cannot use r, but it can lend it to other cells. $I_i(r)$ is initialized to empty.
- $CI_i(r, j)$: a set of cells, which saves the state of $I_i(r)$ when C_i receives C_j's *request* message.

(A) When a cell C_i needs a channel to support a call request, let $temp_i = P_i - U_i - \{r \mid r \in P_i \wedge I_i(r) \neq \emptyset\}$, if $temp_i$ is empty, C_i sets a timer and sends a *request* message to every cell $C_j \in IN_i$; otherwise, it picks a channel $r \in temp_i$ and adds r to U_i. When the call is finished, it deletes r from U_i.

(B) When a cell C_i receives a *request* from C_j, it sends *reply*$(P_i - U_i - \{r \mid r \in P_i \vee (I_i(r) \cap IN_j \neq \emptyset)\})$ to C_j. For any $r \in P_i$, $CI_i(r, j) = I_i(r)$.

(C) When a cell C_i receives all *reply*(S_j) messages or time out, it sets a new timer and does the following:

(C.1) for each r in the system, $B_i = B_i \cup r$ if

- C_i is not using r,
- for any $C_k \in IP_i(r)$, C_i has received a *reply*(S_k) from C_k, and
- for any $C_j \in IP_i(r)$, $r \in S_j$;

(C.2) if B_i is not empty, C_i selects a channel $r \in B_i$ using the underlying channel selection strategy and sends *confirm*(r) messages to all cells in $IP_i(r)$; otherwise, it drops the call.

(D) When a cell C_i receives a *confirm*(r) from C_j, its response depends on the current status of r:

(D.1) If $r \in P_i - U_i - \{r \mid r \in P_i \vee I_i(r) \neq \emptyset\}$, C_i replies with an *agree*(r) and adds C_j to $I_i(r)$.

(D.2) If $r \in U_i$, C_i replies with a *reject*(r).

(D.3) If $I_i(r) \neq \emptyset$, let $temp_i = I_i(r) - CI_i(r, j)$, C_i does the following:
if $temp_i \cap IN_j = \emptyset$
then C_i replies with an *agree*(r) and adds C_j to $I_i(r)$;
else if $\forall C_k \in (temp_i \cap IN_j)$, C_j's *request* has a smaller timestamp than C_k's *request*
then C_i replies with a *conditional_agree*$(temp_i \cap IN_j, r)$ and adds C_j to $I_i(r)$;
else C_i replies with a *reject*(r).

(E) If C_i receives a response to its *confirm*(r) from every cell in $IP_i(r)$ and the following two conditions are satisfied, then the request is successful:

(E.1) Every response is either an agree or conditional-agree.

(E.2) For each *conditional_agree*(S, r) and for each $C_j \in S$, C_i has received an *agree*(r) from some $C_k \in (IP_i(r) \cap IP_j(r))$.

Otherwise, the request is failed. In case of success, C_i uses r to support the call and sends *release*(r) to every cell in $IP_i(r)$ when the call terminates; in case of failure or time out, C_i sends an *abort*(r) to those cells in $IP_i(r)$ from which it has received an agree or conditional-agree message, deletes r from B_i, and then goes to step (C.2).

(F) Outdated *reply*, *agree*, and *conditional_agree* messages are discarded by comparing timestamps. When a cell C_i receives a *release*(r) or *abort*(r) from C_j, it deletes C_j from $I_i(r)$.

Figure 2. The distributed channel acquisition algorithm.

common available channel r in C_{A_4}, C_{A_5}, C_{A_7}, and C_{A_8}, C_{A_4}, C_{A_5}, C_{A_7}, and C_{A_8} are interference primary cells of r relative to C_{H_4} and C_{F_4}; i.e., $IP_{H_4}(r) = IP_{F_4}(r) = \{C_{A_4}, C_{A_5}, C_{A_7}, C_{A_8}\}$. If C_{H_4}'s *request* arrives at C_{A_4}, C_{A_5}, and C_{A_7} earlier than C_{F_4}'s *request*, but arrives at C_{A_8} later than C_{F_4}'s *request*, C_{A_4}, C_{A_5}, C_{A_7}, or C_{A_8} cannot send *agree* messages to both C_{H_4} and C_{F_4} due to the possibility of an interference. Without the use of *conditional_agree* messages, both C_{H_4} and C_{F_4} cannot use channel r. With the help of *conditional_agree* messages, after C_{H_4} gets *agree* messages from C_{A_4}, C_{A_5}, and C_{A_7}, and a *conditional_agree* from C_{A_8}, it can acquire r. Note that C_{F_4} cannot acquire r since C_{A_8} rejects its *request*.

3.2. The channel selection algorithm

Similar to the geometric strategy [2] and the channel selection algorithm in the update approach [9], our channel selection algorithm makes use of the optimal resource planning model defined in section 2. The primary channels for each cell are prioritized. During a channel acquisition, a cell acquires the available primary channel that has the highest priority. If none of the primary channels is available, the cell borrows a channel from its neighbors according to some priority assignment approach. When a cell acquires a channel, it always acquires the channel with the highest priority. When a cell releases a channel, it always releases the channel with the

lowest priority. If a newly available channel has a higher priority than some used channel, an intra-handoff is performed. This can be implemented by assigning a sequence number to each channel and using it to represent the channel priority. With the same example presented in section 2.1, suppose $r1$ is assigned priority $m - 1$ (m is a very large integer) and $r2$ is assigned priority of $m - 2$. Then, C_{A_1} needs to do an intra-handoff (release $r2$ and use $r1$) since its available channel $r1$ has higher priority than that of the used channel $r2$. As a result, C_{H_1} can borrow $r2$.

By assigning a high priority ($m - 1$) to the channel with small sequence number, cells will first use channels with small sequence number, and hence most of the channels with high sequence number are available. As a result, the borrower should try to borrow the channels with high sequence number from others. In order to follow the same channel assignment rule (i.e., acquire the high priority channel), for secondary channels, high sequence number means high priority. Note that it is different from primary channels, where a low sequence number means high priority.

Since the channel status changes frequently, assigning priority only based on the sequence number may not be enough. In our solution, when assigning priority to secondary channels, a cell borrows the channel that has the lowest priority from the "richest" interference neighbors; i.e., the cell with the most available primary channels. The motivation behind this is to reduce the chance that the lender might soon use up its primary channels and have to acquire a secondary channel. Since a channel may have several interference primary cells (three or four in our model), the number of available channels in these interference primary cells may be different. We measure the "richness" by using the interference primary cell which has the lowest number of available channels. More details of the channel selection algorithm can be found in [5].

3.3. The complete channel allocation algorithm

Most of the existing DCA strategies [8,14,15] need up-to-date information to calculate channel priority. This can be easily implemented in centralized algorithms, since the MSC monitors every release and acquisition of the channels, and then it has the up-to-date information. However, in a distributed channel allocation algorithm, due to unpredictable message delay, obtaining the instantaneous global state information is practically impossible. Thus, we can only get the approximately up-to-date information by increasing the message overhead. In order to combine the channel selection algorithm with our distributed channel acquisition algorithm and do not significantly increase the message overhead, we make some modifications to our algorithm.

To make use of locality, a cell does not return the *borrowed* channel immediately after its use. Instead, it keeps the borrowed channel so that these channels can be used when the borrower runs out of channel again. Thus, there are two kinds of borrowed channels in the proposed algorithm: *used-borrowed channel* and *available-borrowed channel.* Used-borrowed channels are counted as used channels. Available-borrowed channels are counted as available channels and can be used again without contacting interference neighbors. When a cell finds that its lender's available channels are less than a threshold η by checking the received *update notification* messages (explain later), it releases the available-borrowed channels from that lender. It should perform an intra-handoff to release the used-borrowed channels from that lender if it is possible. Certainly, the borrower may not be able to return the channel if it cannot find other available channel. By choosing η larger than 0 (such as 5% of the number of primary channels), the borrower has some time to return the channel before the lender runs out of channel. Instead of passively waiting for the notification message from the lender, the borrower checks if its available channels are larger than η' whenever a communication session is over, if so, it releases a borrowed channel since it already has enough available channels.

To reduce the *update notification* message overhead, a cell only notifies the cells that have borrowed channels from it when its available channels are less than η'' ($\eta'' > \eta$). In this way, the message overhead and the intra-handoff overhead can be reduced. However, the borrower may not know the up-to-date information. Knowing the up-to-date information is only helpful when releasing the borrowed channels. Since we want to make use of locality by keeping borrowed channels, and a borrowed channel will be released when its lender's available channels are less than η, it may not be necessary to know the up-to-date information of the lender considering the high message overhead.

To make use of locality, η' should be as large as possible. However, keeping too many borrowed channels may increase the failure rate since other interference cells cannot use them. Certainly, we do not want to make use of locality at the expenses of increasing failure rate. Thus, we choose η' to be a small value. η should be as small as possible so that the borrower can keep the borrowed channel. However, if η is too small, the lender may run out of channel. η'' should be as small as possible to reduce the update notification message overhead. However, a larger value can help the borrower get more up-to-date information. Based on the above considerations and our simulation results, we choose η and η' to be 5% of the number of primary channels, η'' to be 10% of the number of primary channels.

3.4. Reducing the overhead of intra-handoff

Although intra-handoff can improve bandwidth utilization, it also increases the system overhead. In the proposed channel selection algorithm, when a channel r is released, an intra-handoff is needed if there exists a used channel which has higher sequence number than r. In other words, if a new session is admitted while an early communication session is going on in a cell, there will be an intra-handoff when the early session terminates. It is easy to see that the number of intra-handoffs is very high. In our channel priority assignment strategy, a cell always assigns the channel with a small sequence number to the connection request, and lends the chan-

nel with a high sequence number to other cells. As a result, for a cell C_i, if both intra-handoff channels have small sequence numbers, C_i is more likely to have a large number of available channels, and there is a low probability for other cells to borrow the intra-handoffed channels. Thus, the intra-handoffs, between two channels whose channel sequence numbers are all small, are not necessary. After this enhancement, there are still many unnecessary intra-handoffs. In the following, we propose a solution to minimize the number of intra-handoffs without reducing the bandwidth utilization.

An intra-handoff is only necessary when a cell needs to borrow the released channel (by intra-handoff). In the example used in section 2.1, C_{A_1} only performs an intra-handoff between $r1$ and $r2$ when C_{H_1} needs to borrow $r2$ from C_{A_1}. If no cell needs to borrow $r2$, the intra-handoff from $r2$ to $r1$ is unnecessary. The proposed distributed channel allocation algorithm (in figure 2) has two rounds. In the first round, the borrower collects channel status information from its interference neighbors. In the second round, it selects one channel based on the collected information and then confirms the selected channel with the lenders. As a result, the lender does not need to perform any intra-handoff until it receives a *confirm*(r) from the borrower. At this time, the lender checks if it should perform an intra-handoff from channel r to an available channel.

Although the lenders do not perform intra-handoff until the second round of the algorithm, from the borrower point of view, all necessary intra-handoffs should have been done; otherwise, the borrower may not be able to borrow the channel that it should be able to borrow after necessary intra-handoffs (see previous example). Thus, the lender should send modified channel status information (modify U_i and then the channel status in figure 2) to the borrower and perform the real intra-handoff only when the borrower really needs to borrow a channel from it. With the above example, after C_{A_1} receives the channel request from C_{H_1}, it notifies C_{H_1} that $r2$ is an available channel and r_1 is a used channel although it is not true at that time. If C_{H_1} really wants to borrow $r2$ from C_{A_1}, C_{A_1} performs the intra-handoff from $r2$ to $r1$. As a result, $r2$ becomes an available channel and $r1$ becomes a used channel. Note that if C_{A_2} was using $r2$ instead of $r1$, it also needs to do an intra-handoff from $r2$ to $r1$ based on the channel priority rule, before C_{H_1} can really borrow the channel.

Figure 3 shows the pseudocode for the lenders to perform these modifications. For one particular cell, suppose it has j primary channels, which is saved in an array $U[j+2]$, from $U[1]$ to $U[j]$, with "1" representing a used channel and "0" representing an available channel. The first element ($U[0]$) and the last element ($U[j+1]$) of the array are used to control loop termination. To simulate an intra-handoff, the algorithm in figure 2 needs to be modified as follows. In step B, a cell receiving *request* sends a modified U_i (with the pseudocode in figure 3) to the borrower. In step D, when a cell receives a *confirm*(r), it needs to perform an intra-handoff if it is necessary.

```
U[0] = 1; U[j + 1] = 0;
high = j; low = 1;
while (low < high) {
  while (U[low] <> 0) low++;
  while (U[high] <> 1) high-;
  if ((low <> j) && (high <> 0)) {
    U[low] = 1;
    U[high] = 0;
  }
}
```

Figure 3. Simulate an intra-handoff.

The modified intra-handoff strategy minimizes the intra-handoff overhead without affecting the bandwidth utilization. However, it may affect the channel borrowing latency. For example, if the lender has to finish the intra-handoff before replying *agree*, the channel borrowing latency will be increased by the time of an intra-handoff operation. Since the intra-handoff time is very short, many applications can tolerate this small amount of extra delay. As an alternative, the lender can also first reply an *agree* and then do the intra-handoff. Due to the cell-to-cell communication delay, the borrower may be able to use the borrowed channel without interference. In later section, we will see that some borrowed channels may be used for handoff reservation, and then the extra delay or potential interference may be negligible.

4. Integrating distributed channel allocation and handoff management

One of the performance metrics for handoff management is the connection dropping rate (CDR). Ideally, we would like to have zero CDR. However, it requires the network to reserve bandwidth in all cells that an MT might pass through. This is not possible in most cases since the MT's direction is not known as a priori. Moreover, this reservation will severely under-utilize the scarce wireless bandwidth, which will, in turn, cause a high connection blocking rate (i.e., the rate of new connections that will be blocked due to lack of channels). Since it is practically impossible to completely eliminate handoff drops, the best we can do is to provide some form of probabilistic QoS guarantees by keeping the CDR below a pre-specified target value, which is negotiated during the admission control process or set as a system parameter. To achieve this goal, each cell reserves some bandwidth solely for handoff use. The problem is then how much bandwidth should be reserved for handoffs. Reserving a fixed number of channels or making reservation solely depending on prediction may not work well since these approaches may either waste a large amount of wireless bandwidth or fail to reduce the CDR below the target value.

The proposed research aims to address this issue by adjusting the amount of reserved bandwidth based on current network conditions. This is done by measuring the average CDR and the reserved bandwidth usage, and adjusting the amount of reserved bandwidth accordingly. Periodically, or triggered by some events, a cell calculates its CDR. If its CDR is larger (or smaller) than some threshold, the reserved bandwidth is increased (or decreased) by a parameter.

Dealing with heavy congestion. If a cell is heavily congested, there may not be enough bandwidth for handoff reservation. If this situation lasts for an extended period of time due to continued incoming handoffs, the problem becomes more serious because some of the incoming handoffs will be continuously dropped due to the unavailability of reserved bandwidth, triggering further increase of CDR. This, in turn, demands reservation of more bandwidth that does not exist. In order to solve the problem, a cell should block its new connection requests if it finds out that any of its neighbor cells has high CDR and cannot reserve enough bandwidth. However, the solution may block connection requests from the MTs that do not enter the congested cell. Based on the information obtained from the global position system (GPS), the BS may be able to locate the MT and even predict the moving direction of the MT. If GPS is not available, BSs can cooperate with each other to find the location of the MT. Specifically, by measuring the signal strength or the Time of Arrival (ToA) [3,11], the BS can find its distance to the MT. With the cooperation of two other BSs and using basic techniques such as trilateration [3,11], the BS can locate the node by calculating the intersection of three circles. Also, the BS can estimate the MT's moving direction based on a history of observations [6]. With this information, the cell only blocks the requests from the MTs that will enter the congested cell.

To implement this idea, cells need to always communicate with their neighbors to find out if they have enough channels left, which may consume a large amount of bandwidth. To reduce the message overhead, we use the following solution. A cell monitors the *outgoing* handoffs. An outgoing handoff may be dropped if the destination cell does not have enough channels to support the connection. If the outgoing CDR of C_i for the MTs which head toward a destination cell C_j is much higher than the average outgoing CDR of C_i for all six neighbors, C_i blocks β percent of its new connections from the MTs which may go to C_j, where β is system parameter.

Integrating distributed channel allocation and handoff management. Most previous handoff management schemes are based on fixed channel allocation in which a cell cannot borrow channels from other cells. In these schemes, if a cell reserved m channels for handoff, new connections will be accepted only when the number of free channels is more than m. In distributed channel allocation, a cell can borrow channels from other cells. As a result, how to count the number of free channels becomes an issue. In a pessimistic approach, each cell marks the reserved channels as interference channels so that other cells will not count these reserved channels as free channels, and cannot borrow the reserved channels. If each cell does the same thing, the number of free channels which can be borrowed may be reduced. In an optimistic approach, a cell does not mark the reserved channels as interference channels, and cells can still borrow the reserved channels. A connection is admitted as long as it does not violate the guarantee of enough reservation channels even though some of these reserved channels are shared by several neighbors. Since several neighbor cells may count on the same channel as their free channels, it can be a problem when two or more neighboring cells try to acquire the same channel. Also, this approach has high communication overhead, because a cell has to know its neighbor channel status in order to discover the free channels. On the other hand, the optimistic approach can provide more free channels and can be used to accommodate more connections. When an adaptive handoff management scheme is used, since each cell can dynamically adjust the number of reserved channels, the difference between the optimistic approach and the pessimistic approach may be reduced. Furthermore, borrowing channels from neighbors causes some delay, especially when some neighbors are congested. Some handoff applications, such as audio and video, have strict time constraints, so counting on borrowing channels to serve handoff requests may not be a good solution. Thus, we use the pessimistic approach.

Figure 4 shows the adaptive handoff management algorithm which is executed by each cell. When the work load is very high, many cells may not have available channels to be lent to other cells. At this time, sending channel borrowing requests not only wastes networking resources but also wastes the processing power of the BSs. Thus, we use the following approach to further reduce the message overhead. If a cell does not have any channel left, but the number of reservation channels has reached the maximum, the cell C_i sends a *stop* to its neighbors so that these neighbors will not send channel borrowing request to C_i and cells in the interference partition subset of C_i. On the other hand, when the number of available channels grows to a a parameter, C_i sends *resume* to its neighbors so that they can borrow channels from C_i. In this way, the channel borrowing message overhead can be reduced to 0 when the work load is very high.

5. Simulation results

5.1. Simulation parameters

The simulated cellular network is a wrapped-around layout with 12×12 cells. The total number of channels in the system is 720. If a fourth-power law attenuation is assumed [2], the signal to interference ratio is given by $[S/I]_{\min} = [(D_{\min}/R) - 1]^4/6$. With $D_{\min} = 3\sqrt{3}R$, $[S/I]_{\min} \approx 17$ dB, which is a reasonable value in practice. Thus, we choose $D_{\min} = 3\sqrt{3}R$, and then each cell is assigned $720/9 = 80$ channels.

We model the traffic as a *non-uniform* distribution. A cell can be in one of two states: *hot state* or *normal state*. As shown in table 1, a cell spends most of its time in the normal state. A cell in the normal state is characterized by low arrival rate and high inter-handoff rate. On the contrary, a cell in the hot state is characterized by high arrival rate and low inter-handoff rate to picture more arriving new users and prevailing stationary users. Also, the state change rate is assumed to be negative exponentially distributed.

Mobile users are classified as low mobility and high mobility. Under the low mobility category, an MT has the prob-

Data structures.

- $A_i(R_i)$: the number of available (reserved) channels in cell C_i.
- P_i: the CDR of C_i.
- P_{target}: the target CDR of the system.
- P_i^j: the outgoing connection drop rate of C_i for the MTs heading toward cell C_j; C_j represents one of C_i's six neighbors.
- $\overline{P_i}$: the average outgoing connection drop rate of C_i for all six neighbors.
- $\mathcal{I}_1(\mathcal{I}_2)$: the number of reserved channels to be increased (or decreased).

(A) For a new connection which requires m channels, the connection is admitted if $A_i \geqslant m + R_i$; otherwise, use the channel allocation algorithm to borrow $R_i + m - A_i$ channels. The connection is blocked only if not enough channel can be borrowed.

(B) For a handoff requiring m channels, the connection is accepted if $A_i \geqslant m$; otherwise, it is dropped. If $A_i < R_i$, start the channel allocation algorithm to borrow $R_i - A_i$ channels.

(C) After T_{period}, or triggered by a handoff drop, cell C_i calculates P_i, P_i^j, $\overline{P_i}$, and executes the follows.

```
if (there is a handoff drop) {
  if (P_i > P_target) {
    if (R_i < R_max) R_i = R_i + I_1;
    else if (send_i == false) {
      send_i == true;
      send stop to cells in IN_i;
      }
  }
  if (P_i^j > P_i_bar · α) block β percent of connections from the users which
  may head toward C_j;
}
else {
  if (P_i ⩽ P_target) {
    if (A_i > R_i) ∧ (send_i) {
      send_i == false;
      send resume to cells in IN_i;
    }
    if (R_i > R_min) R_i = R_i − I_2;
  }
  if (P_i^j < P_i_bar) stop blocking connections from the users which may
  head toward C_j;
  reset P_i, P_i^j, P_i_bar;
}
```

(D) When a cell C_j receives *stop* message from C_i, it cannot borrow any channel from C_j and then it will not send *request* messages to any cell in IP_j until it receives a *resume* message from C_j.

Figure 4. The adaptive handoff management algorithm.

ability of 0.2 to have a handoff. Under high mobility category, an MT has the probability of 0.8 to have a handoff. There are three kinds of traffics: voice, video, and data, with a required channels of 1, 4, and 2, respectively. A communication connection request has a probability of 0.7, 0.15, and 0.15 to be a voice, video, and data, with a mean service time of 3, 5, and 5 minutes, respectively. When a connection request comes, the BS can correctly predict the direction of the MT with a probability of 80%. We choose P_{target} to be 0.01, $\alpha = 2$, $\beta = 0.5$, $R_{\min} = 0$, $R_{\max} = 16$, which is 20% of the total channel, $\mathcal{I}_1 = \mathcal{I}_2 = 2$. We choose $T_{\text{period}} = 1/P_{\text{period}} \cdot 1/\lambda \cdot 1/(3p_{\text{m}})$.

We considered the connection arrival rate (λ) from 0.1 to 0.6. When $\lambda = 0.3$, the work load is already 100%. Generally, the desirable range of the offered load is less than the link capacity of each cell. It is undesirable to keep a cell overloaded for an extended period of time, and in such a case, the cell must be split into multiple cells to increase the total system capacity. However, cells can get overloaded temporarily. Suppose a mobile user's connection request is blocked once. Then, the user is expected to continue to request a connection establishment until it is successful or the user gives up. This likely behavior of the users will affect the offered load. Near the offered load = 100, the connection blocking rate (CBR) will be larger than 0.1 in most cases. If the connection-blocked user attempts to make a connection about 5 times, the offered load will increase to about 150 in a very short time. Thus, we also model the situation where the offered load is larger than 100%.

5.2. Simulation results

For each call arrival rate, the mean value of the measured data is obtained by collecting a large number of samples such that the confidence interval is reasonably small. In most cases, the 95% confidence interval for the measured data is less than 10% of the sample mean.

5.2.1. Performance of the static reservation

Figure 5 shows the performance of the static reservation approach in terms of CDR, CBR, and bandwidth utilization. Four different configurations are compared, the 20% approach, which reserves 20% of the channels for handoff use, the 10% approach, the 5% approach, and the no-reservation approach ($R = 0$). As shown in the figure, when the number

Table 1
Simulation parameters.

Mean connection arrival rate in a normal cell	λ
Mean connection arrival rate in a hot cell	3λ
Probability of High/low user mobility p_{m}	0.8/0.2
Mean inter-handoff rate in a normal cell	$p_{\text{m}}/60$
Mean inter-handoff rate in a hot cell	$p_{\text{m}}/180$
Mean rate of change from normal state to hot state	1/18000 s
Mean rate of change from hot state to normal state	1/1800 s
Service time per voice/video/data connection	3/5/5 minutes
probability of voice/video/data connection	0.7/0.15/0.15
The number of channels required for each voice/video/data connection	1/4/2

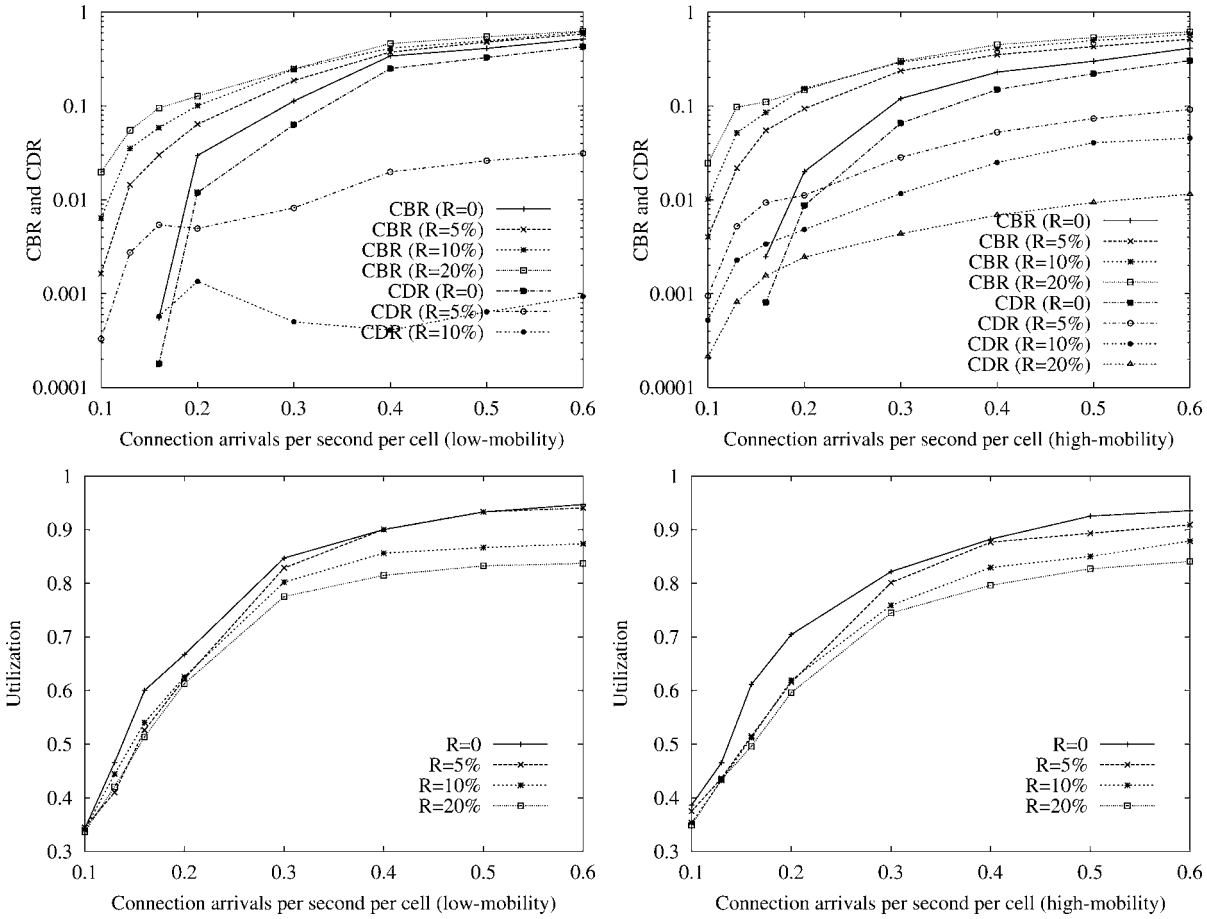

Figure 5. The performance of the static reservation approach.

of reserved channels increases, the CBR increases whereas the CDR drops. For example, with low user mobility and $\lambda = 0.2$, as the number of reserved channel increases from 0 to 20%, the CBR increases from 0.03 to 0.13, while the CDR drops from 0.01 to below 0.0001 (not shown in the figure). Based on the results of the CDR and CBR of different configurations, it is easy to see that the utilization of different reservation approach is different. Generally speaking, the more channels reserved, the less bandwidth utilization it can achieve. For example, with low user mobility and $\lambda = 0.5$, the $R = 0$ approach has a bandwidth utilization of 0.93, whereas the $R = 20\%$ approach has a bandwidth utilization of 0.83.

As shown in figure 5, with high user mobility, reserving 10% of the channel ($R = 10\%$) is enough to keep the CDR below $P_{\text{target}} = 0.01$ when the the connection arrival rate is less than 0.3, but it is not enough for other cases (i.e., $\lambda > 0.3$). Moreover, for low user mobility, the $R = 10\%$ approach seems more than enough (i.e., over-reserved) since the observed CDR is too small compared to $P_{\text{target}} = 0.01$. Note that reserving more than necessary channels may also increase the CBR and reduce the bandwidth utilization. Since the static reservation algorithm only reserves a fix number of channels, it either reserves too many or too less channels, and then it either results in low bandwidth utilization or fails to keep the CDR below P_{target}.

5.2.2. Performance comparisons

In this subsection, we compare the CDR, CBR, and the bandwidth utilization of the static reservation approach (with $R = 10\%$), our algorithm, and the no-outgoing approach, which is our algorithm without considering the outgoing CDR. As shown in section 5.2.1, the $R = 0$ approach and $R = 20\%$ approach are two extreme cases, which optimize one parameter at the cost of the other. Since the $R = 10\%$ approach achieves a balance between the CDR, CBR, and the bandwidth utilization, we only compare our approach with the $R = 10\%$ approach.

As shown in figure 6, our algorithm can keep the CDR below P_{target} and have a relatively high bandwidth utilization. Since our algorithm can dynamically adjust the number of reserved channels, it reserves the proper number of channels and then keeps a good bandwidth utilization. For example, with low user mobility and $\lambda = 0.5$, our algorithm improve the bandwidth utilization by 10% and still keep the CDR below P_{target}. With high user mobility, our algorithm keeps the CDR below P_{target} and it has similar bandwidth utilization as the static reservation approach, but the static approach fails to keep the CDR below P_{target} when $\lambda > 0.3$.

Our algorithm and the no-outgoing approach have similar performance most of the time. However, with high user mobility and $\lambda > 0.4$, the no-outgoing approach is not able to reduce the CDR below P_{target} since the workload is too high

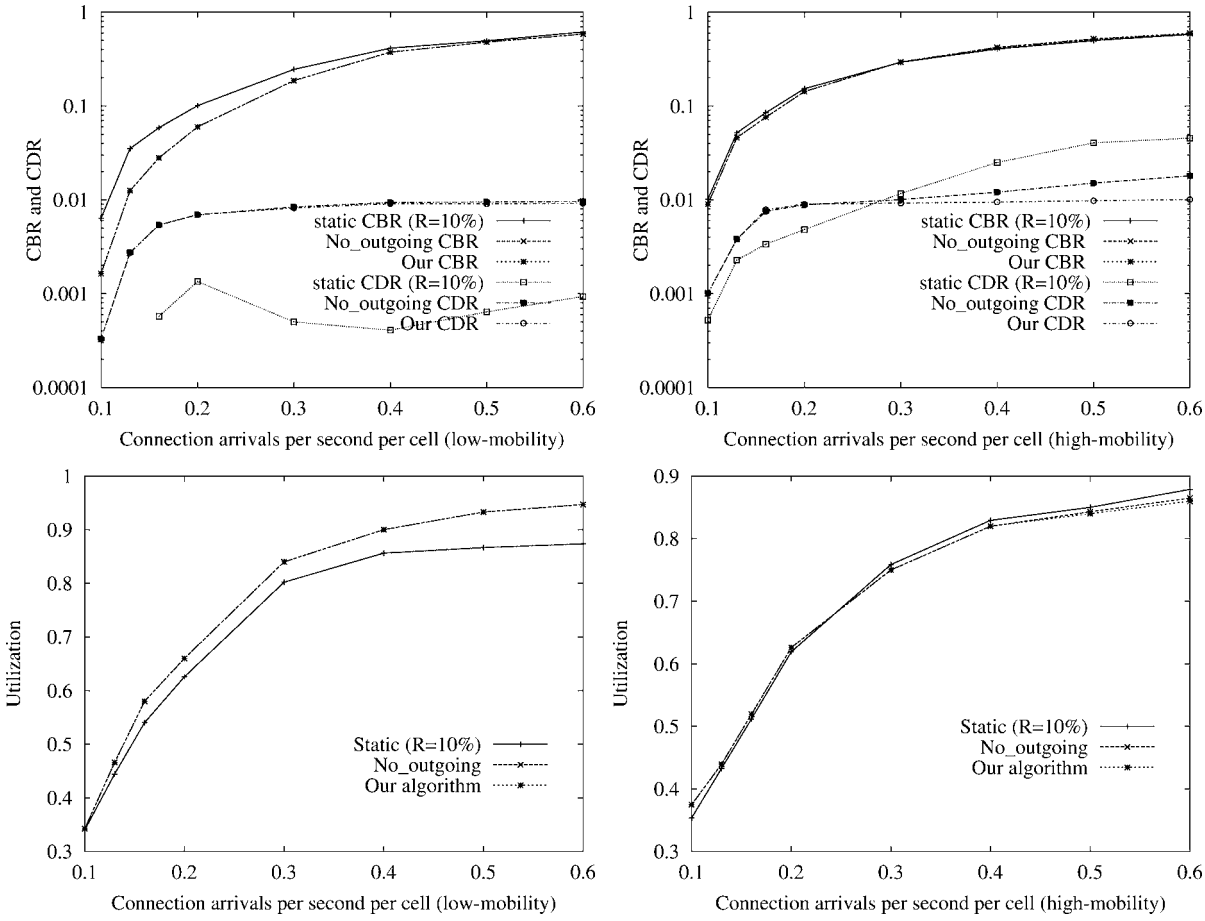

Figure 6. Comparisons of the static reservation approach, our approach, and the no-outgoing approach.

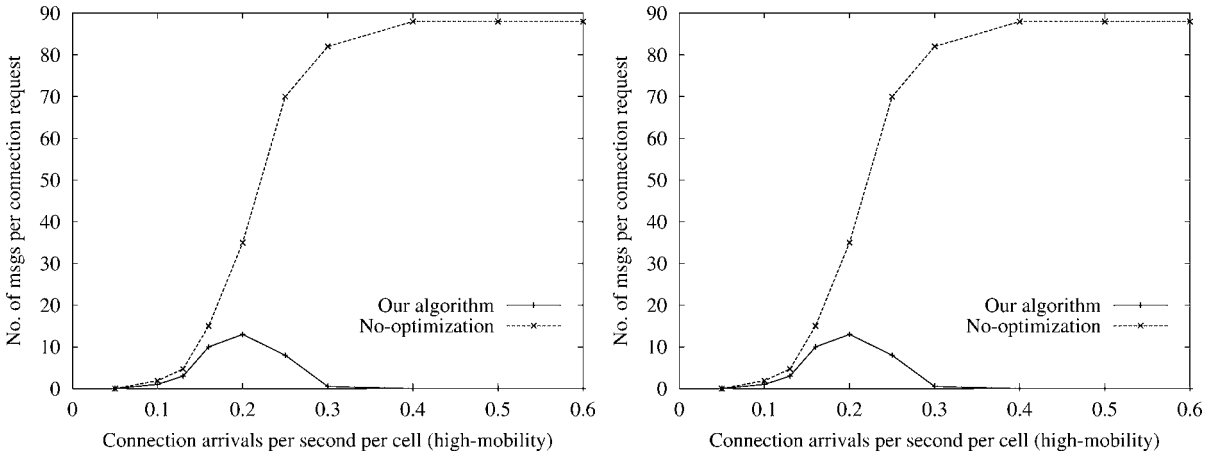

Figure 7. A comparison of the message overhead.

and most of the reserved channels cannot be satisfied. In this case, considering the outgoing CDR can reduce the number of users which heading toward the heavily congested cell, and then further reduce the number of CDR. As a result, our algorithm can still keep the CDR below P_{target} even when $\lambda > 0.4$.

5.2.3. The message overhead

In this subsection, we evaluate the effectiveness of the optimization techniques such as keeping the borrowed channel and using *stop* messages to reduce the unnecessary channel borrows. Without using these optimization techniques, our modified algorithm is referred to as the no-optimization approach.

Figure 7 compares the message complexity of the no-opmization approach and our algorithm. As can be seen, our algorithm has low message overhead compared to the no-optimization approach. In our algorithm, when $\lambda > 0.3$, the workload is more than 100%. As a result, most cells do

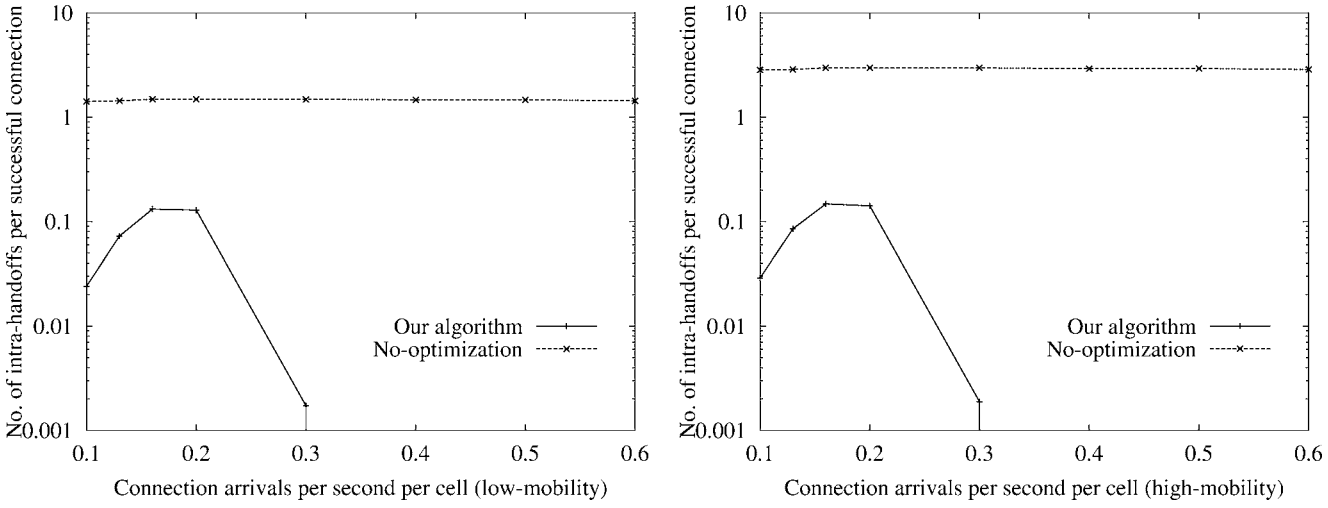

Figure 8. A comparison of the intra-handoff overhead.

not lend channels to other cells by sending *stop* messages to them. Hence, the number of borrowed channels and then the message overhead become 0. On the other hand, in the no-optimization approach, a cell, on receiving a connection or handoff request, starts a channel borrowing process. Since the number of interference neighbors is 30, the message overhead of a channel borrowing is about 60. During a connection, a mobile user may experience a handoff which may trigger another channel borrowing. With high user mobility, each connection may experience two handoffs, and then the message overhead can be as high as $60 \cdot (1+2) = 180$ per connection. However, in figure 7, the message overhead is less than 90. This can be explained by the fact that many connections are blocked when $\lambda > 0.3$, and these connections are not able to handoff to the next cell. Similarly, the message overhead of the no-optimization approach is around 60 with low user mobility.

In our algorithm, the borrower keeps the borrowed channel if the lender has many available channels. By keeping the borrowed channels, our algorithm makes use of the temporal locality and adapts to the network traffic, i.e., free channels are transferred to hot cells to achieve load balance. It can also reduce the message overhead since the borrower may need to borrow channels again just after it returns the previous borrowed channel, and then increases the message overhead. This explains why our algorithm has low message overhead compared to the no-optimization approach even when λ is small.

5.2.4. The intra-handoff overhead

Although intra-handoff can improve the bandwidth utilization, and reduce CDR and CBR, it increases system overhead. In this subsection, we evaluate the effectiveness of our algorithm on reducing the intra-handoff overhead. Figure 8 compares our algorithm with the no-optimization algorithm, which is the same algorithm as our algorithm except that it does not apply the pseudocode in figure 3 to reduce the number of intra-handoffs. As can be seen, the number of intra-handoffs in the no-optimization approach is almost 3 for high user mobility and 1.5 for low user mobility. In the no-optimization approach, when a channel r is released, an intra-handoff is needed if there exists a used channel which has higher sequence number than r. In other words, if a new session is admitted while an early communication session is going on in a cell, there will be an intra-handoff when the early session terminates. With high user mobility, a connection experiences about two inter-handoffs, and then there are about 3 session terminations. With low user mobility, a connection only experiences about one half inter-handoffs, and then there are about 1.5 session terminations. This explains why the no-optimization approach has an intra-handoff of 1.5 for low user mobility and 3 for high user mobility. Note that we only count the number of intra-handoffs for those successful connections.

In our approach, an intra-handoff is only performed when there is a real channel borrowing. Since the number of channel borrowing is reduced to 0 after $\lambda > 0.3$, the number of intra-handoffs is also reduced to 0 after $\lambda > 0.3$. When $\lambda < 0.2$, the number of channel borrowing and the number of intra-handoff increase as the connection arrival rate increases.

6. Conclusions

With the rapid emergence of Internet related applications, providing integrated services such as data, voice, and video through wireless networks has become increasingly important. Since the frequency spectrum available for civilian use is limited, it is a challenge to support QoS using limited spectrum. In this paper, we proposed to integrate distributed channel allocation and adaptive handoff management to provide QoS guarantees and efficiently utilize the bandwidth. First, we presented a complete distributed distributed channel allocation algorithm and proposed techniques to reduce its message complexity and intra-handoff overhead. Second, we integrated the proposed distributed channel allocation algorithm with an adaptive handoff management scheme to provide QoS guarantees and efficiently utilize the bandwidth. Simulation

results showed that the proposed solution can significantly reduce the message complexity and intra-handoff overhead. Moreover, the proposed scheme can improve the bandwidth utilization while providing QoS guarantees.

Acknowledgements

We would like to thank the anonymous referees whose insightful comments helped us to improve the presentation of the paper. This work was supported in part by the National Science Foundation CAREER Award CCR-0092770.

References

[1] A. Aljadhai and T. Znati, A predictive adaptive scheme to support QoS guarantees in multimedia wireless networks, in: *IEEE ICC '99* (1999) pp. 221–225.

[2] A. Baiocchi, F.D. Priscoli, F. Grilli and F. Sestini, The geometric dynamic channel allocation as a practical strategy in mobile networks with bursty user mobility, IEEE Transactions on Vehicular Technology 44(1) (1995) 14–23.

[3] J. Caffery and G. Stuber, Subscriber location in CDMA cellular networks, IEEE Transactions on Vehicular Technology 47(2) (1998) 406–416.

[4] G. Cao and M. Singhal, An adaptive distributed channel allocation strategy for mobile cellular networks, Journal of Parallel and Distributed Computing, Special Issue on Mobile Computing 60(4) (2000) 451–473.

[5] G. Cao and M. Singhal, Distributed fault-tolerant channel allocation for cellular networks, IEEE Journal of Selected Areas in Communications 18(7) (2000) 1326–1337.

[6] S. Choi and K. Shin, Predictive and adaptive bandwidth reservation for hand-offs in QoS-sensitive cellular networks, in: *ACM SIGCOMM'98* (1998) pp. 155–166.

[7] S.K. Das, S.K. Sen and R. Jayaram, A dynamic load balancing strategy for channel assignment using selective borrowing in cellular mobile environments, Wireless Networks (WINET) 3(5) (1997) 333–347.

[8] S.K. Das, S.K. Sen and R. Jayaram, A novel load balancing scheme for the tele-traffic hot spot problem in cellular network, Wireless Networks (WINET) 4(4) (1998) 325–340.

[9] X. Done and T.H. Lai, Distributed dynamic carrier allocation in mobile cellular networks: search vs. update, in: *Proc. of International Conference on Distributed Computing Systems* (1997) 108–115.

[10] S.M. Elnoubi, R. Singh and S.C. Gupta, A new frequency channel assignment algorithm in high capacity mobile communication systems, IEEE Transactions on Vehicular Technology 31(3) (1982) 125–131.

[11] J. Gibson, *The Mobile Communications Handbook* (IEEE Press, Piscataway, NJ, 1999).

[12] D.J. Goodman, Cellular packet communication, IEEE Transactions on Communications 38(8) (1990) 1272–1280.

[13] S. Gupta and P. Srimani, UpdateSearch: a new dynamic channel allocation scheme for mobile networks that adjust to system loads, Journal of Supercomputing 1(17) (2000) 47–65.

[14] H. Jiang and S. Rappaport, Prioritized channel borrowing without locking: a channel sharing strategy for cellular communications, IEEE/ACM Transactions on Networking 4 (1996) 163–172.

[15] I. Katzela and M. Naghshineh, Channel assignment schemes for cellular mobile telecommunication systems: a comprehensive survey, IEEE Personal Communications 3(3) (1996) 10–31.

[16] L. Lamport, Time, clocks and ordering of events in distributed systems, Communications of the ACM 21(7) (1978) 558–565.

[17] K. Lee, Supporting mobile multimedia in integrated service networks, Wireless Networks (WINET) 2 (1996) 205–217.

[18] D. Levine, I. Akyildiz and M. Naghshineh, A resource estimation and call admission algorithm for wireless multimedia networks using the shadow cluster concept, IEEE/ACM Transactions on Networking 5 (1997) 1–12.

[19] S. Lu and V. Bharghavan, Adaptive resource management algorithms for indoor mobile computing environments, in: *ACM SIGCOMM'96* (August 1996).

[20] S. Nanda and D.J. Goodman, Dynamic resource acquisition: distributed carrier allocation for TDMA cellular systems, in: *Proc. of GLOBECOM'91* (December 1991) pp. 883–889.

[21] R. Prakash, N. Shivaratri and M. Singhal, Distributed dynamic fault-tolerant channel allocation for mobile computing, IEEE Transactions on Vehicular Technology 48(6) (1999) 1874–1888.

[22] J. Tajima and K. Imamura, A strategy for flexible channel assignment in mobile communication systems, IEEE Transactions on Vehicular Technology 37(2) (1988).

[23] A. Talukdar, B. Badrinath and A. Acharya, Integrated services packet networks with mobile hosts: architecture and performance, Journal of Wireless Networks (1998).

Guohong Cao received the B.S. degree from Xian Jiaotong University, Xian, China. He received the M.S. degree and Ph.D. degree in computer science from the Ohio State University in 1997 and 1999, respectively. Since Fall 1999, he has been an Assistant Professor of Computer Science and Engineering at Pennsylvania State University. His research interests include distributed fault-tolerant computing, mobile computing, and wireless networks. He has served as an editor for IEEE Transactions on Wireless Communications, and has served on the program committee of various conferences. He was a recipient of the Presidential Fellowship at the Ohio State University in 1999, and a recipient of the NSF CAREER award in 2001.
E-mail: gcao@cse.psu.edu
WWW: http://www.cse.psu.edu/∼gcao